AFHRL-TP-90-70

(2)

AIR FORCE



HUMAN RESOURCES

LABORATORY

# DEVELOPMENT OF THE "CITY OF QUALITY (Coq)" GROUP DECISION SUPPORT SYSTEM



Michael D. Wolfe

DEPARTMENT OF MANAGEMENT
West Virginia University
Morgantown, West Virginia 26506-6101

LOGISTICS AND HUMAN FACTORS DIVISION
Wright-Patterson Air Force Base, Ohio 45433-6503

September 1990

Final Technical Paper for Period June 1989 - August 1990



AIR FORCE SYSTEMS COMMAND
BROOKS AIR FORCE BASE, TEXAS 78235-5601

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely Government-related procurement, the United States Government incurs no responsibility or any obligation whatsoever. The fact that the Government may have formulated or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication, or otherwise in any manner construed, as licensing the holder, or any other person or corporation; or as conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

The Public Affairs Office has reviewed this paper, and it is releasable to the National Technical Information Service, where it will be available to the general public, including foreign nationals.

This paper has been reviewed and is approved for publication.

BERTRAM W. CREAM, Technical Director
Logistics and Human Factors Division

JAMES C. CLARK, Colonel, USAF
Chief, Logistics and Human Factors Division

# REPORT DOCUMENTATION PAGE

*Form Approved*
*OMB No. 0704-0188*



| 1. AGENCY USE ONLY (Leave blank) | 2. REPORT DATE | 3. REPORT TYPE AND DATES COVERED |
|---|---|---|
| | September 1990 | Final Paper - June 1989 - August 1990 |

**4. TITLE AND SUBTITLE**
Development of the "City of Quality (Coq)" Group Decision Support System

**5. FUNDING NUMBERS**
C - F49620-88-C-0053
PE - 62205F
PR - 1710
TA - 00
WU - 18

**6. AUTHOR(S)**
Michael D. Wolfe

**7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**
Universal Energy Systems, Inc.
4401 Dayton-Xenia Road
Dayton, Ohio 45432

**8. PERFORMING ORGANIZATION REPORT NUMBER**

**9. SPONSORING/MONITORING AGENCY NAMES(S) AND ADDRESS(ES)**
Logistics and Human Factors Division
Air Force Human Resources Laboratory
Wright-Patterson Air Force Base, Ohio 45433-6503

**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**
AFHRL-TP-90-70

**11. SUPPLEMENTARY NOTES**
Prepared as final report for 1989 USAF-UES Summer Faculty Research Program.

**12a. DISTRIBUTION/AVAILABILITY STATEMENT**
Approved for public release; distribution is unlimited.

**12b. DISTRIBUTION CODE**

**13. ABSTRACT** (Maximum 200 words)

A general theory for Group Decision Support Systems (GDSS) is here investigated. Mathematical models for multi-person multi-attribute decision making (MPMADM) are developed, and justification for use of a modification of the Taguchi cost function is presented for general use by non-profit organizational MPMADM problems. Total quality and Unified Life Cycle Engineering are just two examples of areas where this group information system may be applied. The House of Quality paradigm is then examined as a framework for the "knowledge acquisition" of the parameters needed for the modified Taguchi loss function, and Hypertext is shown to be an effective platform with which to produce a fully automated version of an integrated, linked series of houses of quality into a City of Quality (Coq) that serves as the basis for our GDSS. New informatics for the production of hypertext systems were developed including self-modifying hypertext documents. As a corollary to the development process, techniques for management and structured development of hypertext projects were created.

**14. SUBJECT TERMS**
decision making
decision support systems
group decision support systems
hypertext
mathematical modeling

**15. NUMBER OF PAGES**
40

**16. PRICE CODE**

| 17. SECURITY CLASSIFICATION OF REPORT | 18. SECURITY CLASSIFICATION OF THIS PAGE | 19. SECURITY CLASSIFICATION OF ABSTRACT | 20. LIMITATION OF ABSTRACT |
|---|---|---|---|
| Unclassified | Unclassified | Unclassified | UL |

AFHRL Technical Paper 90-70 September 1990

# DEVELOPMENT OF THE "CITY OF QUALITY (Coq)" GROUP DECISION SUPPORT SYSTEM

Michael D. Wolfe

Universal Energy Systems, Incorporated
4401 Dayton-Xenia Road
Dayton, Ohio 45432

LOGISTICS AND HUMAN FACTORS DIVISION
Wright-Patterson Air Force Base, Ohio 45433-6503

Reviewed by

Bertram W. Cream, Technical Director
Logistics and Human Factors Division

Submitted for publication by

James C. Clark, Colonel, USAF
Chief, Logistics and Human Factors Division

## SUMMARY

A general theory for Group Decision Support Systems (GDSS) is here investigated. Mathematical models for multi-person multi-attribute decision making (MPMADM) are developed, and justification for use of a modification of the Taguchi cost function is presented for general use by non-profit organizational MPMADM problems. Total quality and Unified Life Cycle Engineering are just two examples of areas where this group information system may be applied.

The House of Quality paradigm is then examined as a framework for the "knowledge acquisition" of the parameters needed for the modified Taguchi loss function, and Hypertext is shown to be an effective platform with which to produce a fully automated version of an integrated, linked series of houses of quality into a City of Quality (Coq) that serves as the basis of our GDSS.

New informatics for the production of hypertext systems were developed, including self-modifying hypertext documents. As a corollary to the development process, techniques for management and structured development of hypertext projects were created.


## Preface

I wish to thank the Air Force Systems Command and the Air Force Office of Scientific Research for sponsorship of this research, and the Logistics Systems Branch of the AFHRL for their cooperation.

Many people were very helpful, and made the summer both enjoyable and productive. Special thanks are due to Capt Hill for his day-to-day support in technical and administrative matters, and to Ms. Edly for other day to day assistance. Thanks also to Mr. Baker for showing me how the CAD system worked to Capt Berry and Mr. Ianni for help with the computer systems, to Capt Painter for showing me Air Force information modeling techniques, to Capt Popken for information about Air Force reliability modeling, to Mr. Smith for listening to some preliminary versions of the results, and to Ms. Stiller for help with the literature search.

Thanks are also due to Ms. Campbell and Mr. Hoffman for their concern and interest in my project.

Finally, the concern of Dr. Duffy and demonstration of her system was greatly appreciated.

## TABLE OF CONTENTS



## FIGURES

# DEVELOPMENT OF THE "CITY OF QUALITY (COQ)" GROUP DECISION SUPPORT SYSTEM

## I. INTRODUCTION

In March 1988, memorandum 30452 was sent from the office of the Secretary of Defense stating that "It is critical at this time that the Department of Defense ... focus on quality as the vehicle for achieving higher levels of performance." A number of Air Force efforts are currently being directed toward this Total Quality Management (TQM). One important facet of these efforts is their need for group technology; many aspects of systems being developed for the Air Force must be considered by many different decision makers.

In particular, the Air Force wants to include reliability, maintainability and other "ilities" in the design phase of projects, rather than addressing these issues after production.

In order to do this, group decision support tools are needed to allow decision makers to assess trade-offs and to communicate their expertise among themselves.

Both a mathematical theory of multi-person multi-objective decision making, and working software are needed for this effort. For some time, the Japanese have had success using a mathematical model by Taguchi for computing optimal system specifications. More recently, a promising technique in use at Mutsubishi's Kobe shipyards was described by Hauser and Clausing. This technique is a very promising approach for capturing the parameters needed by the Taguchi model. An integration of these two approaches into a software package that automates the mechanical details would be of great practical benefit to the Air Force.

## II. OBJECTIVES OF THE RESEARCH EFFORT

My objective for the summer's research was the study of group decision support aids for Concurrent Engineering (CE).

I initially performed an extensive review of relevant literature, both journal articles and relevant reports,with a goal of developing a system to support multi-person, multi-attribute decision making (MPMADM) in support of Quality Function Deployment (QFD) and CE. The review of MPMADM included a taxonomy of such systems and the mathematical theory underlying them. I next reviewed articles and reports on concurrent engineering and quality, including English descriptions of the Taguchi model, the theory of (Deming, 1986), and techniques for developing Hypertext prototypes.

To actually implement the system, I began by outlining the mathematical theory of MPMADM. The specific version chosen for the implementation was the team approach. The specific formulation of the MPMADM problem is based strongly on the Taguchi model, as described in (Elmaghraby et al., 1986). The "House of Quality" paradigm as described in (Hauser and Clausing, 1988) was then chosen as the framework to acquire the parameters required by the Taguchi model.

In order to implement the "House of Quality," I investigated Hypertext as a potential software vehicle. Using Hypertext, I designed a linked ensemble of "Houses of Quality," leading to a tool best described as a "City of Quality (Coq)", a tool that allows the decision makers to communicate their concerns among themselves, that provides a record of decisions made and rationales behind those decisions, that encourages quantification of goals and objectives, and that encourages hierarchical decision making.

I next implemented a very preliminary Hypertext prototype of the system as a "Proof of Concept," and as a guide to the sorts of systems which might be most useful to Air Force decision makers. This prototype is a model "City of Quality" (Coq) system which consists of a linked series of "Houses of Quality."

## III. MULTI-PERSON MULTI-ATTRIBUTE DECISION MAKING (MPMADM)

### Difficulties in MPMADM

Multi-person multi-attribute decision making (MPMADM) is the problem with which real-world decision makers are faced. Mathematically, the problem may be represented as an n-person game in which the decision makers are to solve the following problem:

$$\min_{\mathbf{x}} \mathbf{L}(\mathbf{x},\theta) \qquad (3.1)$$

$$\text{S.T. } \mathbf{x} \in \Omega$$

The vector $\mathbf{x}$ is the **decision vector**, to be set by the several decision makers acting in conjunction; the vector $\theta$ is a random vector, called the **state of nature**; the set $\Omega$ is the **technology set** of feasible decisions; $\mathbf{L}(\mathbf{x},\theta)$ is the (possibly non-linear) **loss function**, which, for this problem, is a matrix of losses, with $L_{ij}(\mathbf{x},\theta)$ the loss of attribute j incurred by decision maker i. See Fig. 3.1.

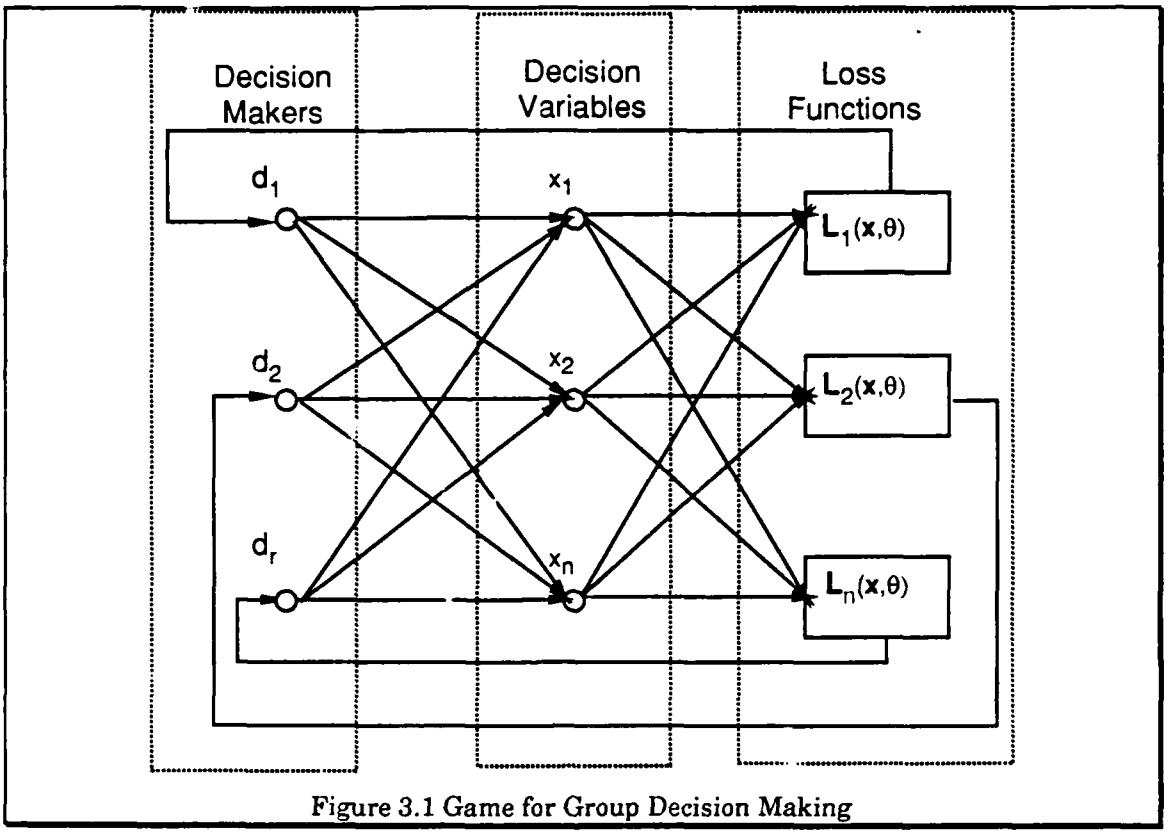


Figure 3.1 Game for Group Decision Making

Note that, for this general form of the game, any decision maker may potentially influence any component of the decision vector. This is the basic game theoretic formulation, as in (Luce and Raiffa, 1957)

In addition, not shown in Figure 3.1, is an **information function** $\zeta(\theta)$. $\zeta_i(\theta)$ is the information available to decision maker i. One of the decision variables available to each decision maker is the possibility of communicating $\zeta_i(\theta)$ to one or more other decision makers. The loss function includes the costs of this communication.

The problem (3.1) is usually approached by attacking each of its constituent parts; unfortunately, not only are all the parts, in general, intractable, our goal is solution of the overall problem.

To summarize the difficulties encountered in trying to solve equation (3.1) observe:

1) the equation does not really make sense, even for a single decision maker, if each $\mathbf{L}_i(\mathbf{x}, \theta)$ is a vector: there is no ordering on $\Re^n$, $n \geq 2$, which is consistent with the usual topology, hence no single methodology for multi-attribute decision making can be a prescriptive standard;

2) even if every decision maker were to have a scalar loss function of some sort, if these functions are all different, there is, by Arrow's Impossibility Theorem (Arrow, 1963[1]), no logically consistent method to combine them into a group decision, except under special circumstances;

3) if the group somehow settles on a scalar group loss function, $l(\mathbf{x},\theta)$, there is no agreement on optimal methods for dealing with uncertainty;

4) if, finally, the group agrees on a deterministic scalar loss function, (e.g. expected value $\phi(\mathbf{x}) \equiv E(l(\mathbf{x},\theta))$ this is, in general, non-linear and non-convex, and the set $\Omega$ is, in general, a non-convex mixed integer domain, so the resultant optimization problem:

$$\min_{\mathbf{x}} \ \phi(\mathbf{x}) \qquad (3.1a)$$

$$\text{S.T. } \mathbf{x} \in \Omega$$

has no efficient means of (exact) solution. In the absence of some rather strong assumptions which enable the problems 1—4 to be circumvented, a system to support the various aspects of group decision making may not be programmed to provide a general solution algorithm for MPMADM, but will, rather, be an information and decision support system, making the cost of communication (including intertemporal communication) as low as possible, and providing access to as many of the available individual decision support tools as possible.

Overview of general solution approaches to MPMADM

In the absence of good, canonical methods for solving the problems (1)—(4) above, a large number of approaches have been proposed. These may be summarized as follows:

(1) Multi-attribute decision making. Disregarding $\theta$ for now, assume that $L \equiv \{L(x)\}$ is the set of all multi-attribute losses available to a single decision maker. A **multi-attribute loss** is represented as a vector in ordinary Euclidean space $\Re^n$, i.e., it is assumed that each attribute may be individually quantified. In addition, assume that the decision maker has some rational preference ordering on $L \subseteq \Re^n$. Then the following holds:

[1]Arrow actually refers to this as the Possibility Theorem. Modern usage, however, refer to the theorem as either the Impossibility Theorem, or as the Dictator Theorem.

**Proposition 1.** If L is a countable ordered set, then there is an order-preserving mapping

$$\Phi: L \rightarrow Q \tag{3.2}$$

where $Q$ is the rational numbers.

For n>2, $\Re^n$ is a topological vector space, but $Q$ is an ordered field, and it is tempting (though not justified by decision theory) to use those field properties. In addition, $\Phi$ is usually extended from L to $\Re^n$. The simplest method of doing this is the usual linear multi-attribute utility theory. A vector of parameters $\pi \in \Re^n$ is statistically estimated such that

$$\pi^T l = \Phi(l) \quad \forall l \in L \tag{3.3}$$

induces the correct ordering on L. Unfortunately, if $|L| > n$, (3.3) is overdetermined, and, has no solution, except in a least-squares sense. In general, such a solution will not agree with the decision maker's original ordering only for a subset of L. A non-linear variation uses the decision-makers preferences for hypothetical lotteries to fit a smooth curve or piecewise linear function to $\Phi$. In (Currim and Sarin, 1984) efforts along these lines led to a linear $\Phi$ that fit 63% of decisions made, and to a non-linear $\Phi$ that fit 80% of the decisions, but in no case were they able to find a functional form for $\Phi$ that fit all decisions with any positive degrees of freedom.

One may accept as a solution a choice of $\Phi$ that covers "most" of L, or claim that (3.3) (or a non-linear variation) is a prescriptive model for multi-attribute decision making.

A less controversial approach is that of Pareto-optimality. This imposes the natural partial order on $\Re^n$:

$$\mathbf{x} \leq \mathbf{y} \text{ if } x_i \leq y_i \ \forall i \tag{3.4}$$

$$\mathbf{x} < \mathbf{y} \text{ if } \mathbf{x} \leq \mathbf{y} \text{ and } \mathbf{x} \neq \mathbf{y}.$$

then a point $l \in L$ is **non-Pareto optimal** with respect to minimization if

$$\exists l' \in L \ni l' < l \tag{3.5}$$

A point is **Pareto optimal** if it is not non-Pareto optimal. Basically, a point is Pareto optimal if no feasible point can reduce the loss associated with one attribute without increasing the loss associated with another attribute. The drawback to Pareto optimality is that many non-equivalent points will generally be Pareto optimal, and the theory of Pareto optimality offers no method for choosing between Pareto optimal points. Work by Storbeck

(oral communication) indicates that decision makers may not even eliminate from consideration points that are not Pareto optimal, judging the lack of optimality to reflect noisy data, rather than actual inferiority.

However, any information system to support multi-attribute decision making must , at an absolute minimum, inform the decision makers about the Pareto-optimality of solutions under consideration.

(2) Group decision making. A first simplification which partially circumvents Arrow's theorem is to reduce from the group decision making problem to the **distributed decision making problem**, in which each decision maker has complete control over some of the decision variables, as in Figure 3.2.

Arrow's theorem does not apply, since each decision maker is allowed to select a preference for the value of some decision variables independent of the choices of the other decision makers. Unfortunately, game theory has examples of multi-person games with no reasonable notion of solution, so a further simplification is commonly made to **team decision making** in which there is a single loss function for all players, as in Figure 3.3. This is the case studied in, for example, (Marschak and Radner, 1972).

Each player i in this game has access to an **information function** $\zeta_i(\theta)$ and may **signal**, or communicate $\zeta_i(\theta)$ to team members $j \neq i$ at some cost $\chi_{ij}$, which is included in the loss function. This signalling is explicitly indicated in the figure.

One method of improving the expected value of the loss function is to reduce the cost of communication, which is the goal of Coq.

(3) Uncertainty. The random vector $\theta$ may represent **risk**, the situation where $\theta$ has a known distribution (or at least a known expected value $E(\theta)$ and variance matrix $Var(\theta)$) or it may represent **uncertainty**, where the distribution is unknown. Since the mean and variance represent two independent attributes (assuming all other moments may be disregarded in the analysis), this represents a return to multi-attribute decision making. The methods most commonly used here are similar to (1) except that the function $\Phi$ is either **expected utility**, or, as a further simplification, **expected value**, in the case of risk. A further refinement proposed by (Charnes and Cooper, 1961) is to minimize the expected value of the loss, subject to a (set of) constraints which proscribe unacceptably large losses:

$$\min_{\mathbf{x}} E(L(\mathbf{x}))$$

$$\text{S.T. } P(L(\mathbf{x})) \geq \mathbf{b}) \leq \mathbf{e} \qquad (3.6)$$

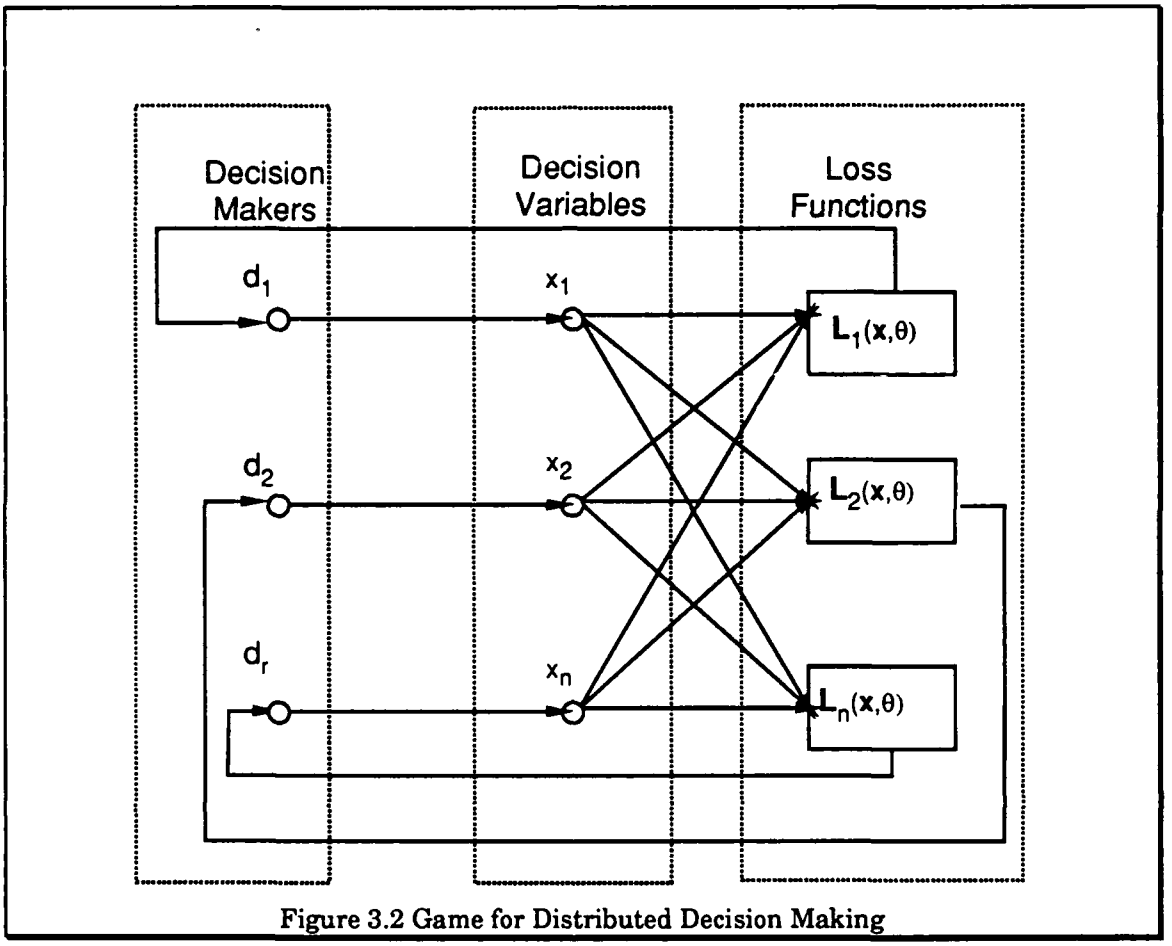

Figure 3.2 Game for Distributed Decision Making

This is called **chance-constrained programming** in (Charnes and Cooper, 1961).

The traditional method for dealing with uncertainty, as opposed to risk, is to impose some distribution, usually, in the case of complete uncertainty, a uniform prior, or, where some information is available, the maximum entropy prior.

(4) Non-linear optimization. Given the assumptions that must be made for (1)—(3) a form for the resultant objective functional $E(\Phi(L(\mathbf{x},\theta)))$ may be selected in such a way that the optimization is amenable to whatever solution procedures are on hand, whether they be heuristic or algorithmic, numerical or symbolic.

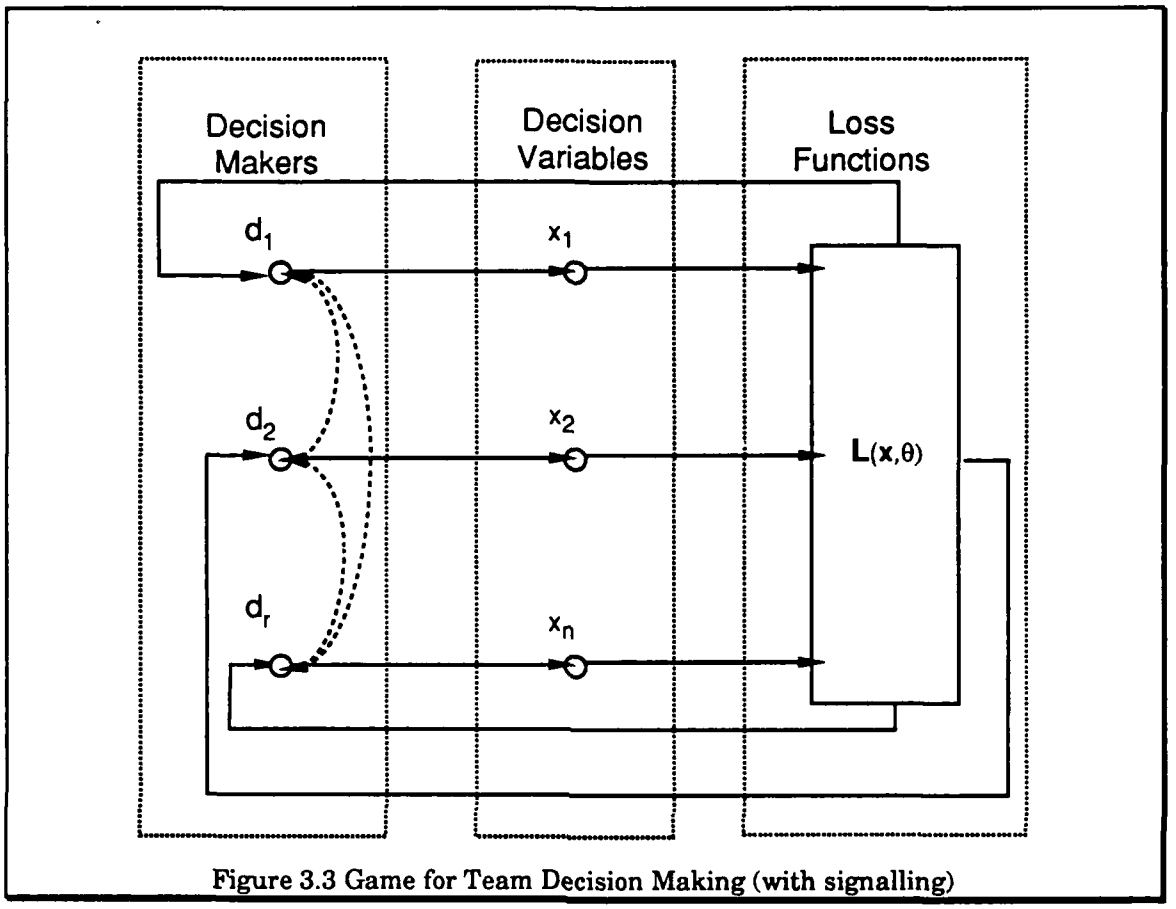


Figure 3.3 Game for Team Decision Making (with signalling)

## IV. TEAM BASED MPMADM WITH MODIFIED TAGUCHI LOSS FUNCTIONS

Based on (1)—(4) above, it is necessary to develop appropriate **L** and $\Phi$ for QFD. For the sake of notation consistent with the literature, introduce the **production process**. Decision makers are given a decision vector **s**, which may also be interpreted as input to the production process. The process then produces quantifiable outputs

$$\mathbf{x} = \mathbf{p}(\mathbf{s},\varepsilon) \tag{4.1}$$

where $\varepsilon$ is a random vector, $\mathbf{x} \in \Re^n$ is a random vector, $\mathbf{s} \in \Re^m$ is determined by the decision makers in a way that minimizes the loss $\mathbf{L}(\mathbf{x},\theta)$ associated with **x** and the random state of the world, $\theta$. In general, unfortunately, producers do not know **p** except for some set $S \subseteq \Re^n$ (often a small, finite set). This is an impediment to minimizing $\mathbf{L}(\mathbf{x},\theta)$.

For non-profit agencies, the usual notation of maximizing a profit function does not make sense. The non-profit agency would seem more like a consumer, maximizing mission effectiveness subject to a budget constraint, solving the problem

$$\max_{\mathbf{s}} \; Eff(\mathbf{x}) \tag{4.2}$$

$$S.T.\; C(\mathbf{s}) \leq B$$

Assuming $Eff(\mathbf{x})$ to be an isotonic function of B, (4.2) is equivalent to

$$\max_{\mathbf{s}} \; Eff(\mathbf{x}) \tag{4.2'}$$

$$S.T.\; C(\mathbf{s}) = B$$

This becomes a problem when distributed decision making is used. Here (4.2) is hierarchically decomposed into a series of subproblems, with subvectors $\mathbf{s}_i \in \Re^{n_i}$ of $\mathbf{s}$ and $\{B_i\} \ni \sum_i B_i = B$.

The coordinator's problem is then

$$\max_{\{B_i\}} \; Eff(\mathbf{x}) \tag{4.3a}$$

while the subordinates' problems are

$$\max_{\mathbf{s}_i} \; Eff(\mathbf{x}_i)$$

$$S.T.\; C(\mathbf{s}) = B_i \tag{4.3b}$$

where $\mathbf{x}_i \equiv \mathbf{p}_{|\Re^{n_i}}(\mathbf{s}_i, \varepsilon_i)$.

Exact solution of (4.3) would lead to a good global solution; however, while solution of problem (4.3b) is often straightforward, solution of (4.3a) is usually quite difficult in which case the solutions to (4.3b) may lead to erroneous results. A more robust formulation of the non-profit producers' problem is given by:

$$\min_{\mathbf{s}} \; C_p(\mathbf{s}) + C_c(\mathbf{x}) \tag{4.4}$$

where $C_p(\mathbf{s})$ is the **producer's cost** of selecting the decision variables $\mathbf{s}$ and $C_c(\mathbf{x})$ is the consumers' cost associated with production of output $\mathbf{x}$. Taguchi's first contribution is the notion that a producer should minimize total cost to the organization (or nation) rather than maximizing its own profit. In his original formulation, as described in (Elmaghraby et al., 1986),

however, Taguchi omitted the producer's cost. Equation (4.4) has been called the extended Taguchi cost function in (Tse and Cralley, 1988).

In addition to the general form (4.4), Taguchi also specified the specific form to be used for $C_c(\mathbf{x})$: the consumer is presumed to have a target $\tau$ for $\mathbf{x}$. Then

$$C_c(\mathbf{x}) = \sum_i w_i (x_i - \tau_i)^2 \tag{4.5}$$

where $\mathbf{w}$ is a vector of weights to be applied to the deviations. The scalar loss function is then

$$L(\mathbf{s}) = C_p(\mathbf{s}) + E(C_c(\mathbf{x})) \tag{4.6}$$

$$= C_p(\mathbf{s}) + \mathbf{w}^T\sigma^2 + \mathbf{w}^T(\bar{\mathbf{x}} - \tau)) \tag{4.7}$$

where $\bar{\mathbf{x}}$ is the mean of $\mathbf{x}$ and $\sigma^2$ is the variance of $\mathbf{x}$.

This formulation is particularly felicitous, as it is readily amenable to distributing the decision processes among the team: beginning with the final consumer's requirements $\mathbf{x}_0$, the producer may establish a decision vector $\mathbf{s}_1$ that minimizes (4.6). The components of $\mathbf{s}_1$ may then require production, so the process is repeated with $\mathbf{s}_1$ in place of $\mathbf{x}$ and $\mathbf{s}_2$ in place of $\mathbf{s}$. The global nature of (4.6) ensures that, if each sub-problem is solved to minimize $L(\mathbf{s}_i)$ the resulting solution will be optimal. In addition, the formulation is unconstrained. Thus, this approach is also recommended by (Ross, 1988).

Depending on available software and on the nature of the problem, the simple form (4.5) may be modified without affecting the main thrust of the analysis. One may use, for example, a piecewise quadratic rather than a simple quadratic function, absolute values rather than quadratics, or the satisficing formulation as described by (Charnes and Cooper, 1961) rather than an optimizing formulation.

We assume that $C_p(\mathbf{s})$ is readily available: i.e., that for any choice of the decision variable $\mathbf{s}$, the costs of production are known. The team information function is described by the estimates each of the team members has for $\varepsilon$, $\theta$, $\tau$, $\mathbf{p}$, and $\mathbf{w}$, $\hat{\varepsilon}_i$, $\hat{\theta}_i$, $\hat{\tau}_i$, $\hat{\mathbf{p}}_i$ and $\hat{\mathbf{w}}_i$. The assumption that this is a team decision problem implies that all members of the team share the scalar loss function (4.6) and the function of the GDSS is to obtain the best possible group estimates for $\varepsilon$, $\theta$, $\tau$, and $\mathbf{w}$ based on the individual estimates, i.e., the GDSS provides a **group function**

$$G: (\hat{\varepsilon}_i;\ \hat{\theta}_i;\ \hat{\tau}_i;\ \hat{\mathbf{p}}_i;\ \hat{\mathbf{w}}_i)(\mathbf{s}) \rightarrow (\hat{\varepsilon};\ \hat{\theta};\ \hat{\tau};\ \hat{\mathbf{p}};\ \hat{\mathbf{w}})(\mathbf{s}) \quad \forall \mathbf{s} \in \Omega \tag{4.7}$$

which takes the matrix of individual estimates and computes the group estimate. An optimal group function may be obtained as a consequence of:

> **Proposition 2.** Let $\mathbf{y}$ be a random vector, $\hat{\mathbf{y}}_i$ a matrix of estimates of $\mathbf{y}$, and $\varepsilon_i = \mathbf{y} - \hat{\mathbf{y}}_i$ a matrix of random errors. If the $\varepsilon_i$ are unbiased, then $\hat{\mathbf{y}}$, the minimum variance unbiased linear group estimate of $\mathbf{y}$ is

$$\hat{\mathbf{y}} \equiv \frac{\mathrm{Var}^{-1}(\varepsilon_i)\hat{\mathbf{y}}_i}{\left|\mathrm{Var}^{-1}(\varepsilon_i)\right|} \tag{4.8}$$

As discussed in (Pyeatt and Wolfe, 1989) this solves the problem of reconciling expert judgements from a panel of experts, even when the experts' answers are correlated. It does not, however, solve the problem of expert bias, which is a matter for further research.

Application of Proposition 2 requires that we obtain an estimate of $\mathrm{Var}^{-1}(\varepsilon_i)$ based on prior decisions and results. Hence the system must maintain such a database.

## V. CITY OF QUALITY TEMPLATE FOR GDSS

The framework selected for collecting the data identified in §4 is based on the model observed at Mitsubishi and described by (Hauser and Clausing, 1988). The original work requires the decision makers responsible for design, development, and production of a system to fill out four "Houses of Quality." Our extension automates the links between the four houses, to produce an integrated system we call the City of Quality (Coq). The Coq:

1) Assists decision makers to establish all requirements. In the notation of §4, Coq reduces the chance that an important target, $\tau_i$, in (4.5) above is omitted—i.e., given an incorrect weight $w_i = 0$; this is listed as a serious problem in (De Vera and others, 1988)

2) Provides a framework for setting formal weights and targets, rather than leaving the decision process fuzzy and intuitive;

3) Maintains a record of how trade-offs were resolved;

4) Reduces the probability that a customer requirement fails to be addressed in the final design.

The system that does this is shown in Figure 5.1, which shows the basic House. The left side of the house, labeled "Customer Attributes," contains the customer requirements; a column is provided for their relative weights. It is designed to encourage a hierarchical approach to the requirement definition, which will be discussed further below. The "attic"

(just below the roof) labeled "Engineering Characteristics" in conjunction with the bottom of the house is used to list quantitative, engineering specifications, target values, importance, estimated difficulty and costs. Also provided are spaces for comparison with existing designs.

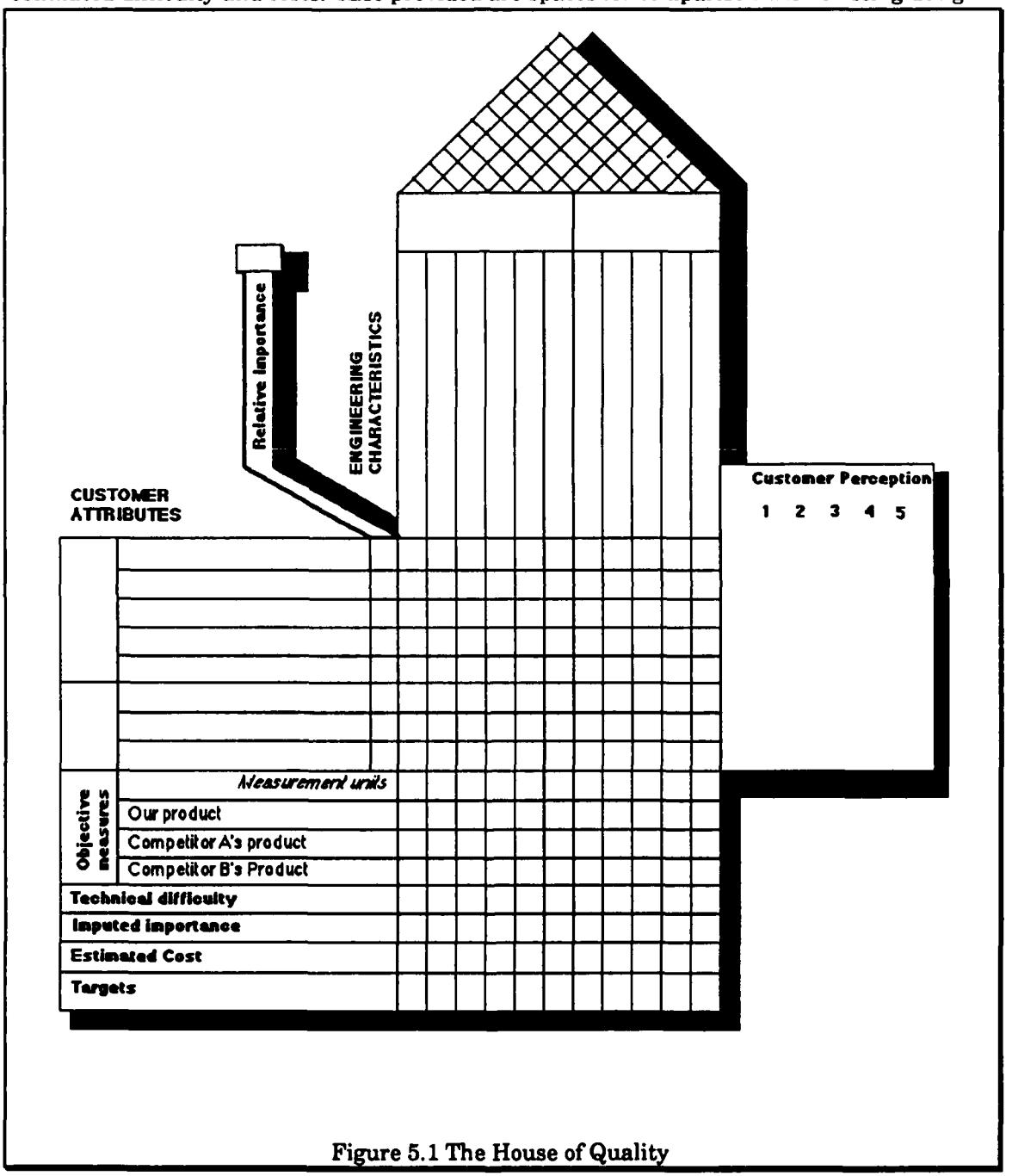


Figure 5.1 The House of Quality

The central portion of the house provides space to indicate which engineering specifications address (or degrade) a customer requirement. The roof indicates the synergies

and conflicts among the engineering specifications. Finally, a graph on the right hand side of the House indicates competitive position.

This decomposes the group decision problem into the various cognitive parts, and provides the support for (1)—(4) above.

Support for Trade-off Analysis

The design of the Coq system is intended to facilitate the difficult problem of trade-off analysis.

Minimizing the loss function (4.6) is complicated by the fact that the production function $\mathbf{p}:\Re^n \rightarrow \mathbf{X} \subseteq \Re^m$ may not be onto; in particular, it is possible that $\tau \notin \mathbf{X}$. In addition, $\frac{\partial x_i}{\partial s_j}$ may be related to $\frac{\partial x_k}{\partial s_j}$, in such a way that selection of $\mathbf{s}$ to improve target $\tau_i$ degrades target $\tau_k$. Selection of that value of $\mathbf{s}$ which minimizes (4.6) must thus be done either informally or formally. Support for formalizing the decision process is supported by (Keeney and von Winterfeldt, 1989) and this is provided by the central matrix of the House and by the roof of the house.

For example, in (Garvin, 1987) eight primary customer requirements are identified for all systems: 1) Performance; 2) Features; 3) Reliability; 4) Conformance; 5) Durability; 6) Serviceability; 7) Æsthetics; and 8) Perceived quality. A technical specification that addresses one of these will tend to degrade one or more of the others, as very high performance items have a (justifiable) reputation for reduced reliability and durability.

We emphasize that all decision makers, with or without a decision support system, must make these trade-offs, consciously or unconsciously. Within our framework, ignoring reliability in favor of performance consists of putting a preemptive weight $w_i$ on the performance characteristic. The Coq system allows this choice on the part of the decision maker, but makes it explicit.

It will also be the case that the engineering specifications may exhibit synergy or antagonism, and this is explicitly indicated on the roof of the House. Taking this synergy into account may result in significant cost savings, while explicitly acknowledging the antagonistic factors may prevent an unanticipated product failure.

This basic support for explicit trade-off analysis is the model from Mitsubishi described by Hauser and Clausing. What Coq provides is access to sophisticated decision support and modeling tools to facilitate the process.

Since Coq is a very low cost system, it is available to a wide range of decision makers who may formerly have had no access to GDSS software, and who may not be familiar with

the concept of trade-off analysis. Hence an interface is provided to provide a great deal of support to the decision makers.

Communication is facilitated by the fact that users of Coq are prompted for relative weights, for indications of which customer attributes are addressed or degraded by which technical specification, and for the synergies and antagonisms among the technical specifications. The user may specify these relationships in a form tailored to the user's cognitive mode, and the system will then display the results in a common form, for the benefit of other group members.

Where simple models and formulae are appropriate (e.g. unit conversions) or where a database of physical constants is needed, the software provides "slots" where these may be inserted. Complex models developed in FORTRAN or other conventional languages are also completely accessible. The calculations needed by designers are not only available from within Coq, they become part of the House, and hence may be reviewed by other members of the group.

Intertemporal Communication Support

In development of a large, complex system, the design process is spread out over a period of months, or, more often, years. Many of the decision makers responsible for production and maintenance at the end of this process will have had little or no contact with the decision makers responsible for the initial statement of need and preliminary design. While the original Hauser and Clausing article indicated that this could be done with linked Houses, as shown in Figure 5.2, their design left a lot of tedious details to be addressed by manual, paper methods. Coq is designed to automate these details, and to provide additional functionality as well.

Coq is designed for any decomposition of the system life cycle into phases, although for illustration the stages suggested by (Hauser and Clausing, 1988) are used. At each interface, data from the previous stage are automatically transferred to the next stage; in addition, as each requirement at stage i is addressed by a technical specification, this information is relayed back through the previous stages. The result is a system that ensures that some specification addresses each customer requirement. The original system suggested that this could be done manually; in practice, large systems might have hundreds of requirements, and it is unlikely that decision makers would take sufficient care in checking each row for an indication that the requirement is addressed; however, this guarantee may be programmed into a suitable software package which can produce an exception report and bring the failing to the immediate attention of the decision maker.

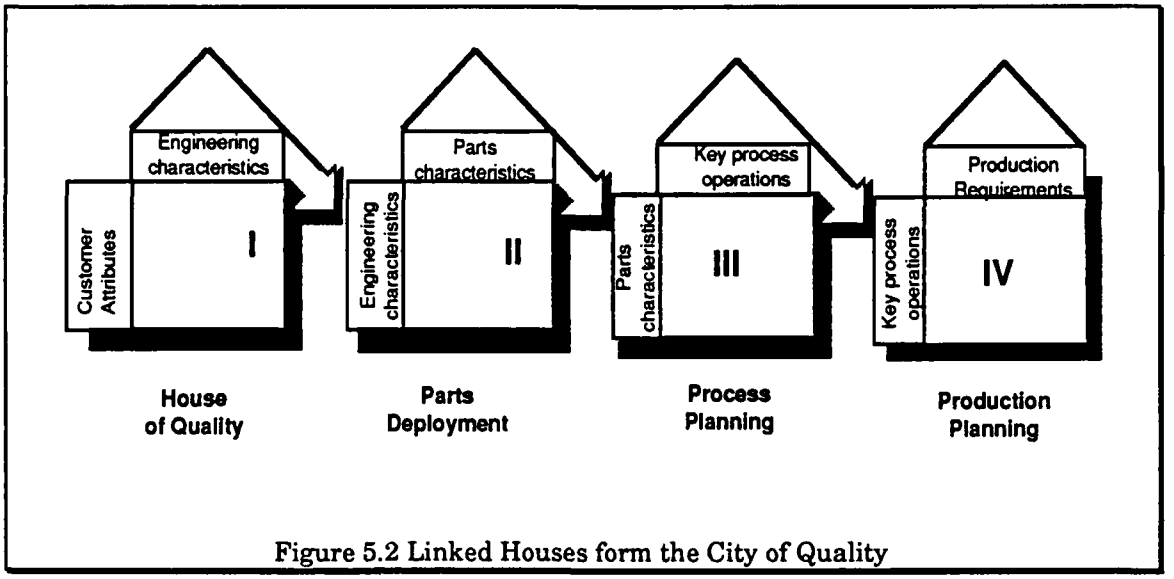


Figure 5.2 Linked Houses form the City of Quality

From a more theoretical perspective, provided that the decision space **S** is compact, the formulation of the loss function (4.6) is a real valued function, and hence must have a global minimum, i.e., there is a "right answer" to the problem as stated. In fact, there are two "right answers:" the answer that minimizes expected loss, given the limited information at the time the decision is taken, and the answer that minimizes actual loss if the random factors $\varepsilon$ and $\theta$, the production function **p**, the optimal targets $\tau$ and the appropriate weights $w_i$ for each target $\tau_i$ be known. Rational decision makers will always take the decision that minimizes the expected loss of (4.6); when this differs from the posterior optimal solution, the error is due to erroneous estimation of one of these five vectors. From Proposition 2, this estimate may be optimized if the system has an estimate of the variance of the error of the estimates made by the decision makers. In order to estimate this variance accurately, a record must be kept of prior decisions and results.

This has tended not to happen (Hill, Duffy, oral communication). It is not possible for an information system to enforce usage; however, Coq is designed to be as user friendly as possible, to encourage decision makers to record decisions, rationales for those decisions, and the engineering level formulæ that support those decisions.

## VI. HYPERTEXT

### Overview of Hypertext

The concept of hypertext is credited to Vannevar Bush, while the word is credited to Ted Nelson by (Smith and Weiss, 1988) and by (Vaughan, 1988). Pure **hypertext** is a non-

linear form of writing. Ordinary written documents may be visualized as a linear graph as in Figure 6.1. Hypertext, on the other hand, allows an arbitrary network as in Figure 6.2. Topologically speaking, the linear graph of ordinary text can be embedded in one dimensional space, while the network of hypertext requires at least two dimensions, and possibly three.

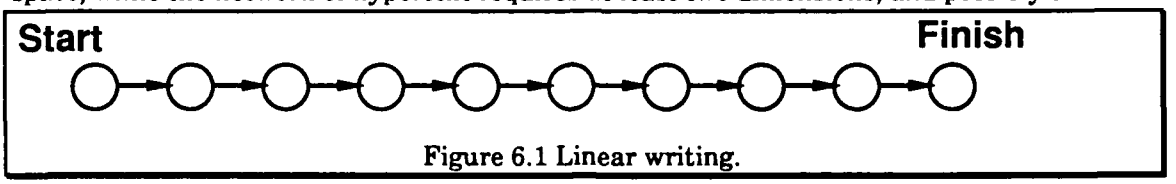


Figure 6.1 Linear writing.

Each node in the network represents some text, and is, perhaps, best visualized as a page in an encyclopedia or instruction manual. The reader, given a book or other paper document, is forced to move sequentially through the text, with random access tedious, and the time required for such access interrupting the reader's train of thought.

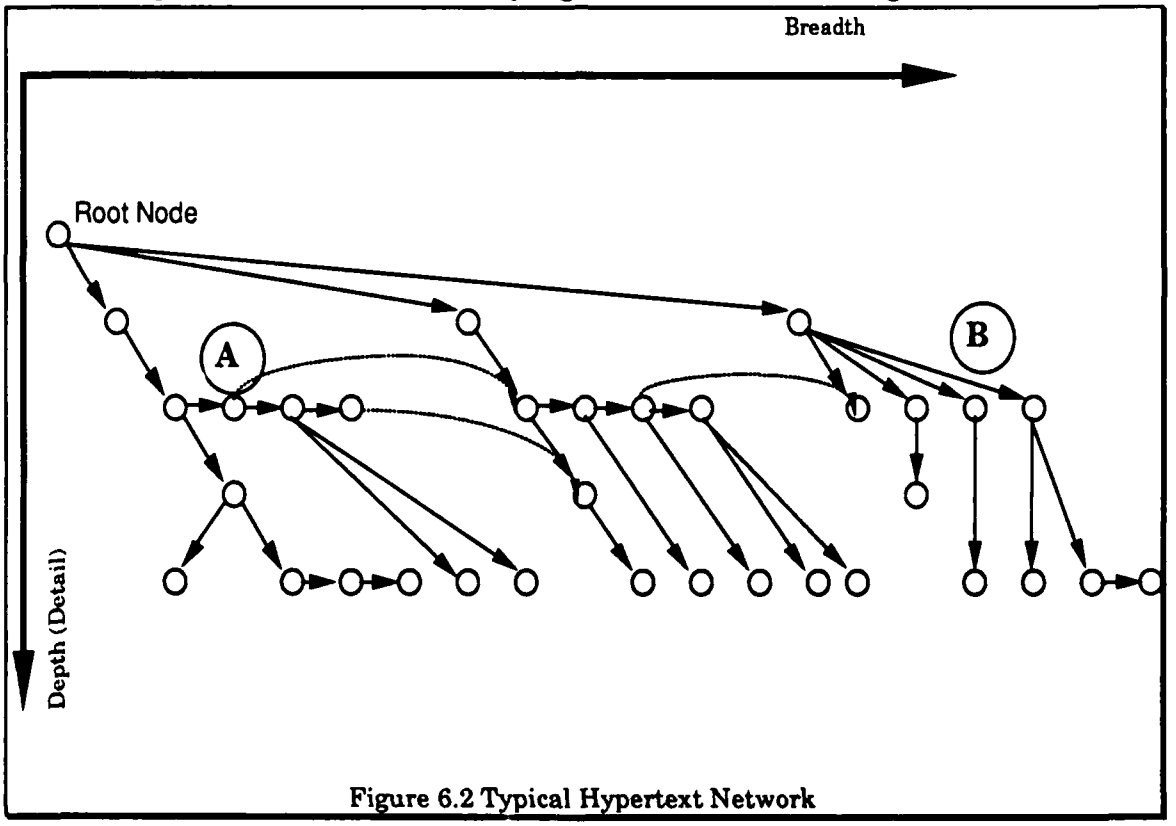


Figure 6.2 Typical Hypertext Network

The horizontal dimension represents breadth; the vertical dimension represents detail. Arcs connect related nodes. A straight horizontal arc indicates a simple narrative; for example, the four nodes at A connected by three straight horizontal arcs might represent four pages in an article. If two nodes at the same vertical level are connected by a straight horizontal arc from node i to node j, node i is called a **predecessor** of node j, while node j is

called a **successor** of node i. If a node has a successor, this is the default link at the node: this is the natural path if the user does not need additional detailed information, and does not wish to follow up on cross-references. There can be only one successor arc from each node, as is obvious from the (Euclidean) geometry, since only one straight line arc may extend directly to the right from a given node.

The straight line arcs directed diagonally downward depict the hierarchy of information: lower nodes represent details about higher nodes. For example, at B in the Figure, there are no successor arcs, and hence no narratives continued across nodes. The reader may peruse deeper regarding a subject, but there is no direct connection between the nodes at a level.

As is usual in network terminology, if node i at a higher level is connected to node j at a lower level, node i is called a **parent** of node j, while node j is called a **child** of node i.

> Caution. For persons familiar with network and hierarchical databases, this is not the same network structure: nodes are not so much entities (in the sense of network databases) but pages of text, and the parent/child relation here is unrelated to the formal parent/child relation between data items in the network and hierarchical data models.

Unlike successors, a node may have any number of children, as the information on a single page might have a number of details (in regular text, footnotes.)

Finally, the curved arcs represent relationships between nodes that are more serendipitous than that of predecessor/successor or parent/child, and indicate cross-references that might occur. These are called **sibling** arcs, whether they connect nodes on the same level or different levels, since depth in different branches of the network (measured as distance from the root node) is not really comparable. Nodes connected by sibling arcs are called **siblings**. Again, a node may have many siblings, as a page of text may have many cross-references. Sibling arcs, unlike parent/child arcs, are usually bi-directional.

As in normal network terminology, a node with no children, siblings or successors is called a **leaf** node. A node to which only leaf nodes are attached is a **twig** node.

All nodes and arcs are created by the author of the hypertext document. For the reader, access to a node is primarily limited to traversing a connected path to that node. On opening a hypertext document, the reader is placed at the root node, and may proceed along any available arc. The canonical term for reading a hypertext document is **browsing**. The arcs are directed, and it is not possible to traverse an arc backward; however, it is possible to back up, i.e., to retrace a path already traversed. The distinction is clear when a node has

multiple parents: one may back up to the parent via which the node was accessed, but may not access any other parent directly.

In some implementations, the reader may have access to a search command which allows direct access to the next node in the document (or possibly all nodes) satisfying the reader's search criterion. Access to the search function may be under the author's control, and may be present at some nodes, absent at others, depending on how structured the author wishes the exposition of the document to be. The most common implementation of this search function is a simple string search, although a few implementations of hypertext provide sophisticated search facilities. The arcs explicitly established by the author are called **hard links**; additional **soft links** are created when the user does a search, as hypertext remember the paths taken, and allows them to be retraced.

The advantage to hypertext is that it is almost as easy to follow the parent/child arcs and sibling arcs as it is to follow the normal successor path, making it possible to readily proceed at the desired level of detail, to check any of the cross-references provided by the author, and, by using the string searching feature, to check other cross-references that occur to the user while browsing the document.

In its pure form, however, hypertext does not provide facilities for communication of data between nodes: this requires the more advanced product, hypermedia.

## Hypermedia

While Bush probably envisioned pure hypertext, the storage of information on electronic media encourages much more creative methods for communicating that information. At a minimum, graphics are added if the hardware supports them; at the same conceptual level, digitized sound may be played at a node. This introduces a second network structure, an **inheritance net**, which is independent of the hypertext network. In displaying information, identical (or very similar) graphics and sounds may be mixed with different text in displaying information to the end user. Rather than re-creating the shared graphical/sound resources, a node may **inherit** these resources from another node. Again, if an inheritance arc exists between node i and node j, we call node i a **parent** and node j a **child.**

The hypermedia concept is not, however, limited to static text and graphics. In its most general form, each node may be a complete program, which executes when the node is accessed. Many implementations of hypertext/hypermedia include a programming language. Nodes in which the underlying program is written in this "native" programming language

may be general nodes; if the program is an external program, the node it represents must be a leaf node.

In this general form, hypermedia is an underlying structure connecting available electronic resources in whatever manner the author feels is most suitable for the presentation of information. Nodes could include CAD/CAM programs, animations, explanations, etc.

The limitation of pure hypertext mentioned above, that data cannot be transmitted between nodes, does not apply when each node is a program, which may store and retrieve data from shared resources either in core memory or from secondary storage facilities.

The Hypercard® Implementation of Hypertext: Capabilities and Limitations

Because Hypercard was the first mass produced and readily available implementation of hypertext, is is often mistakenly assumed that Hypercard≡Hypertext. Hypertext had, in fact, been implemented in prototypical forms on workstations for many years before Hypercard appeared, and is currently available in many forms on many platforms. Hypercard is merely the most accessible implementation available at this time.

In Hypercard, each node is displayed as a card the size of a Macintosh Plus screen; on other platforms, the same black and white card is displayed. In addition to ordinary text, support is provided for bitmapped graphics, sound, arithmetic operations, and simple programming. A simple inheritance structure is also provided: cards (nodes) with common graphical features and placement of text share a common **background**, from which they inherit these common features. Support for more complex inheritance schemes is only provided by the recursive programming language, which allows authors to implement such inheritance for themselves. No support is provided for vector graphics or color.

For use in a group setting, it is desirable for users to be able to add comments to nodes in a way that limits accidental interference, a feature called **non-disruptive annotation** in (Akscyn, McCracken, and Yoder, 1988) who observe that this is an important part of their system[2]. Again, this may be programmed by the author of a Hypercard document, but is not provided as part of the standard system.

In actual implementations, there is a physical design as well as a logical design. In Hypercard, a physical file containing one or more cards (and one or more backgrounds) is called a **stack**. In terms of performance, switching to a stack not recently accessed is rela-

---

[2]Their system was originally called ZOG, but the engine, separated from the specifics of the original application, is now a general purpose hypertext engine called KMS.

tively time consuming. In addition, the search function, mentioned earlier, is designed to search on a single stack. Access to this function is controlled by the author; however, should the author wish to give the reader the ability to search over several separate stacks, each search must be provided separately.

Standard Hypercard provides limited facilities for use over a LAN; however, these facilities may be purchased separately by members of Apple's development group, APDA. Existing tools are limited to stack locking, rather than card locking. An ideal system would allow two or more users to simultaneously view a card and attach comments, but this does not seem possible with Hypercard at this time, so sequential multi-user access will be required. In addition, the system designer must make the trade-off between the greater multi-user accessibility of a system with many small stacks vs. the enhanced performance of a system with a few large stacks.

The facilities are also primitive in the area of security. Security issues in a multi-user database include security from inadvertent damage to other users' data, and deliberate damage to other users' data. Protection from the latter is especially difficult. The Hypercard programming language makes sophisticated security programming possible, but, again, programming is necessary, as security features are not built into the program.[3]

One feature present in Hypercard which does provide excellent security, however, is the Oracle® interface. This enables a Hypercard user to store all (sensitive) data on an Oracle database either on the Hypercard platform machine, or on a mainframe computer, with all the security thereby available. By having cards that contain no data of their own, but merely access it from a multi-user mainframe, some of the other multi-user problems can be circumvented, but this requires programming the access sequence to the mainframe, as well as mainframe database programming.

The final limitation of Hypercard is performance. Given the low-end platform available and the fact that it is an interpreted language, access to large databases with complex programs executing at each node can result in unacceptable response times. This does not, however, reduce its appeal for prototyping, and as a terminal vehicle for moderate sized data bases with only simple processing at most nodes.

---

[3]A prompt for passwords and a simple encryption scheme are provided, but the steps to be taken after prompting and receiving either a valid or invalid password must be programmed by the author.

## Design methodology for Hypertext

Rapid prototyping Development of any information system involves user interface issues and data processing issues. Experience shows that the former tend to be more difficult and as important as the latter, but tend to be grossly underweighted in the initial design process. All the problems in systems life cycle that this (and related software systems) are intended to address are, in fact, present in the life cycle of the software system itself. With traditional life cycle methodologies, decisions made early in the design cycle, when designers are faced with the greatest uncertainty about the system, have the greatest impact on the cost and usability of the system.

The key problem, called the "Feigenbaum Bottleneck" in (Michie, 1983) and (Duke, Nijssen, and Twine, 1988) is the production of a formal specification of the problem which corresponds with the users' needs. Initially, users and system designers often have difficulty communicating just what those needs are, and the prototype has been found to be an effective medium for this communication. The idea is to produce a working version of the product using a very high level language, one that facilitates modifications and rapid development of user interfaces. This prototype may evolve into a production product; more often, the performance limitations of the very high level language make the prototype unacceptable in a production mode. In the latter case, the prototype forms a formal specification for the production version—the user "signs off" on the prototype, and the desired appearance and functionality of the production system are now clearly established. All that remains is to implement the system in a high performance, perhaps more portable version.

Design standards for Hypertext. Various organizations have proposed standards for structured software development. Few apply to hypertext projects. In typical hypertext projects, data flow and program flow diagrams are meaningless: the data do not flow, nor does the program. The hypertext document is static, with the dynamics supplied by the user. Even a paper user's manual would largely re-create the information in the hypertext document, but in a less accessible format.

Of the hypertext projects reported to date in the open literature, none have provided design standards/methodologies comparable to those established for other software projects. While much research remains to be done in this area, a few points have been established.

1) Supporting documents for a hypertext project include:

a) the **logical layout**, showing the nodes and arcs;

b) the **inheritance structure chart** of the nodes;

c) the **physical layout** of the nodes on the various storage media used;

d) conventional documentation for any programs which execute at the nodes (omitted if the nodes are just text, no program;)

2) The structured design process should include:

Before any actual programming

a) preparing the logical layout chart;

b) preparing the inheritance structure chart, to minimize redundancies;

c) preparing conventional documentation, if appropriate;

d) review of the above documents;

Followed by

e) implementation of the preliminary prototype;

f) preparation of the physical layout chart;

g) refinement (go to a)).

3) The implementation process should have controls that ensure uniformity at each node, so that the identical symbols produce parallel effects.

The following conventions were chosen for the logical structure chart, in addition to the basic conventions of successor arcs, parent/child arcs, and sibling arcs described earlier: initially, the tentative, preliminary name of every node (and brief description, if not clear from the name) is filled in using an italic font. As a node is implemented, the italics are replaced with roman type. Important nodes are indicated with boldface type. This way, the structure chart also serves as a dynamic project management tool.

The reserved word "Local" is used to indicate nodes that serve to support another node, "Help" nodes, for example, and the facility that allows a reader to attach comments to the node. A structure chart following these conventions will be displayed in a later section.

Synergy between Hypertext and Coq template. The basic idea of a Coq, as shown in Figure 5.2, suggest some sort of network. From the root, an image derived from Figure 5.2, the network branches to the four houses. Less obvious is that the basic House of Quality shown in Figure 5.1 has a hierarchical decomposition, well suited to a network: this is the decomposition discussed in §5, into

1) customer attributes, weights, etc;

2) technical specifications, cost, importance, etc.;

3) the matrix indicating which specifications address which requirement, and to what extent;

4) the roof indicating trade-offs between the various technical specifications.

In addition, there is interest in encouraging a hierarchical approach to the acquisition of customer attributes and technical specifications, and this is particularly appropriate to the hypertext network: a node is assigned for customer attributes. This has, as children, primary attributes. These children have, as their children, nodes for secondary attributes, etc.

Beyond this is the ability to attach notes and comments to any part of a house. The intersection of the row for an attribute and the column for a specification forms a node. This node has, as children, separate nodes corresponding to each of the decision makers in the group responsible for the design. This nodes then have as children nodes with formulæ used to establish the specification from the attribute and with comments from the decision makers explaining their rationale for trade-offs made. This sort of network, operating interactively, is "readily" implemented in hypertext, but with difficulty using conventional programming.

## VII. PRELIMINARY DESIGN OF THE COQ HYPERTEXT SYSTEM

### Description of the Design

The first step in the design of a Hypertext system is establishment of the logical layout chart, described in §4. The root node of any Hypercard document is called "Home" and provides access to a variety of applications. These could have been deleted; however, Coq tries to keep as many tools and facilities as possible to encourage decision makers to use the system and avoid the "weak link" problem where one key decision maker does not use the GDSS and renders the entire system ineffective. The root to the Coq tree is Figure 5.2. Overall, the top level of the tree looks like Figure 7.1:

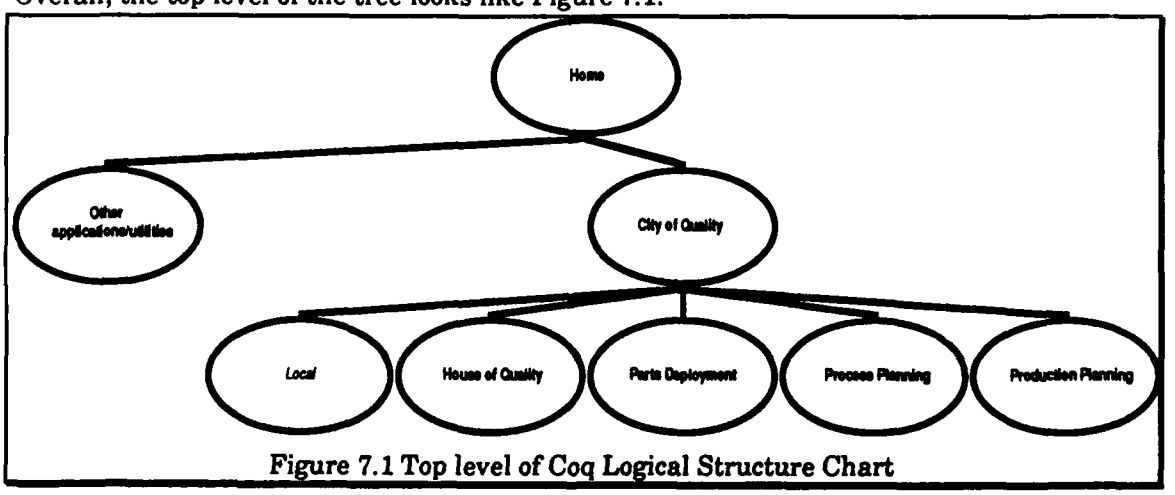


Figure 7.1 Top level of Coq Logical Structure Chart

The four non-local children of the City of Quality node all look like Figure 5.1 when accessed (except for titles). They function in similar ways, with one important exception: in

the basic house, much of the task to be accomplished is the acquisition of customer attributes, while in the other houses (parts, process, production) the left hand side is automatically filled in by the system with the information from the previous house. The structure for these houses is shown in Figure 7.2.

The most complex structure, however, is the initial House of Quality, because this must:

1) Acquire the customers' attributes, which are somewhat vague, and more qualitative than quantitative; then

2) Translate these into formal, quantitative, engineering-level specifications.

Later houses must do the more structured process of translating formal specifications from one arena to another, which is much easier than the unstructured (or semi-structured at best) process of translating informal requirements into technical specifications. The top-level logical structure of the House is shown in Figure 7.3.

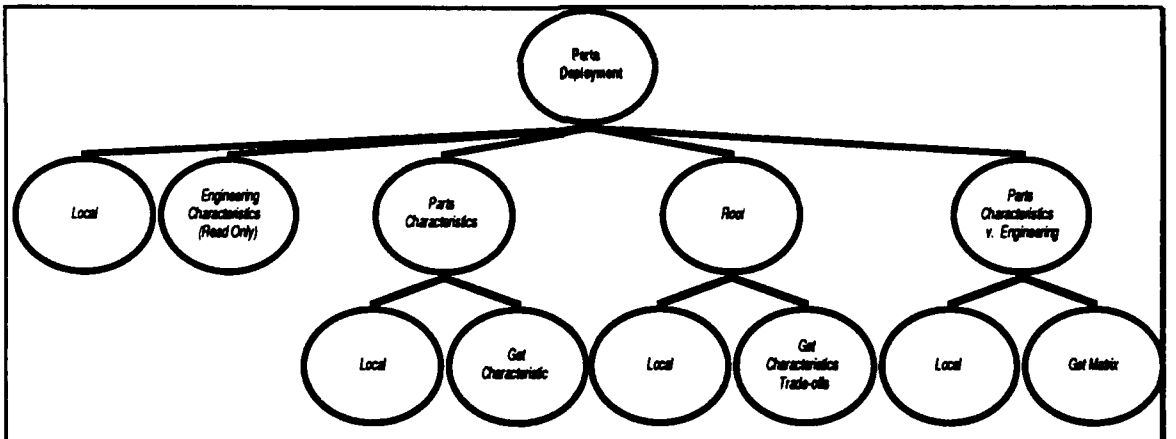


Figure 7.2 Logical Structure for Parts Deployment Subnet

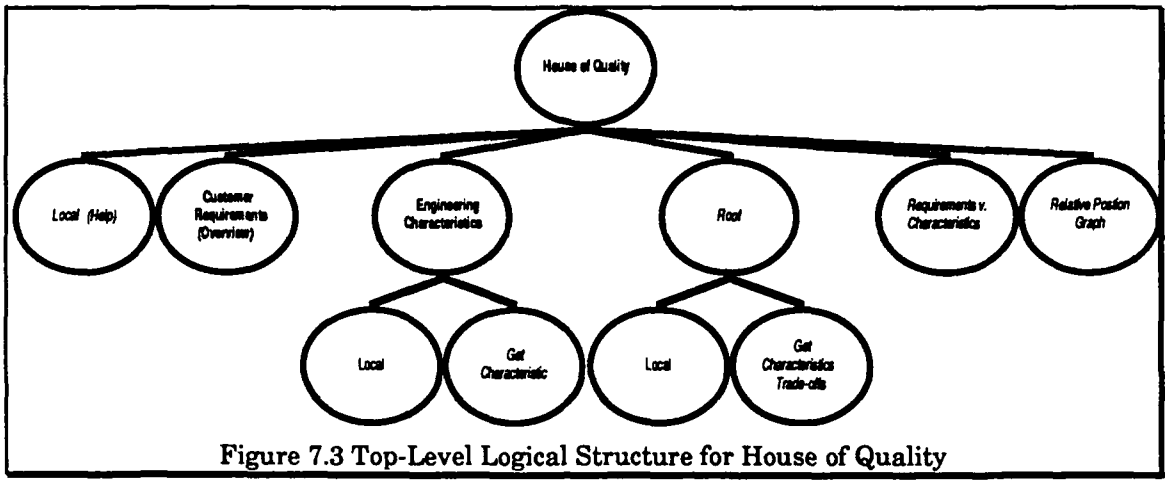

Figure 7.3 Top-Level Logical Structure for House of Quality

The subsystem for acquisition of customer requirements is a separate, almost stand-alone module for GDSS. It is designed to implement the first stage of multi-attribute decision making, by acquiring the matrix of required attributes and weights for each decision maker involved. This matrix may then be used in any number of models: the system has a very open architecture that allows co-ordinators to select any of the pre-installed linear options, or to add options of their own.

The logical structure chart for this subsystem is complicated by the fact that it is dynamic. This is a new concept in Hypertext: the development of Coq involves basic research, and new information technologies are being developed. Initially, the node for acquiring the customer attributes is a twig node (its only subnode being the local node with help information). The node has the appearance shown in Figure 7.4.

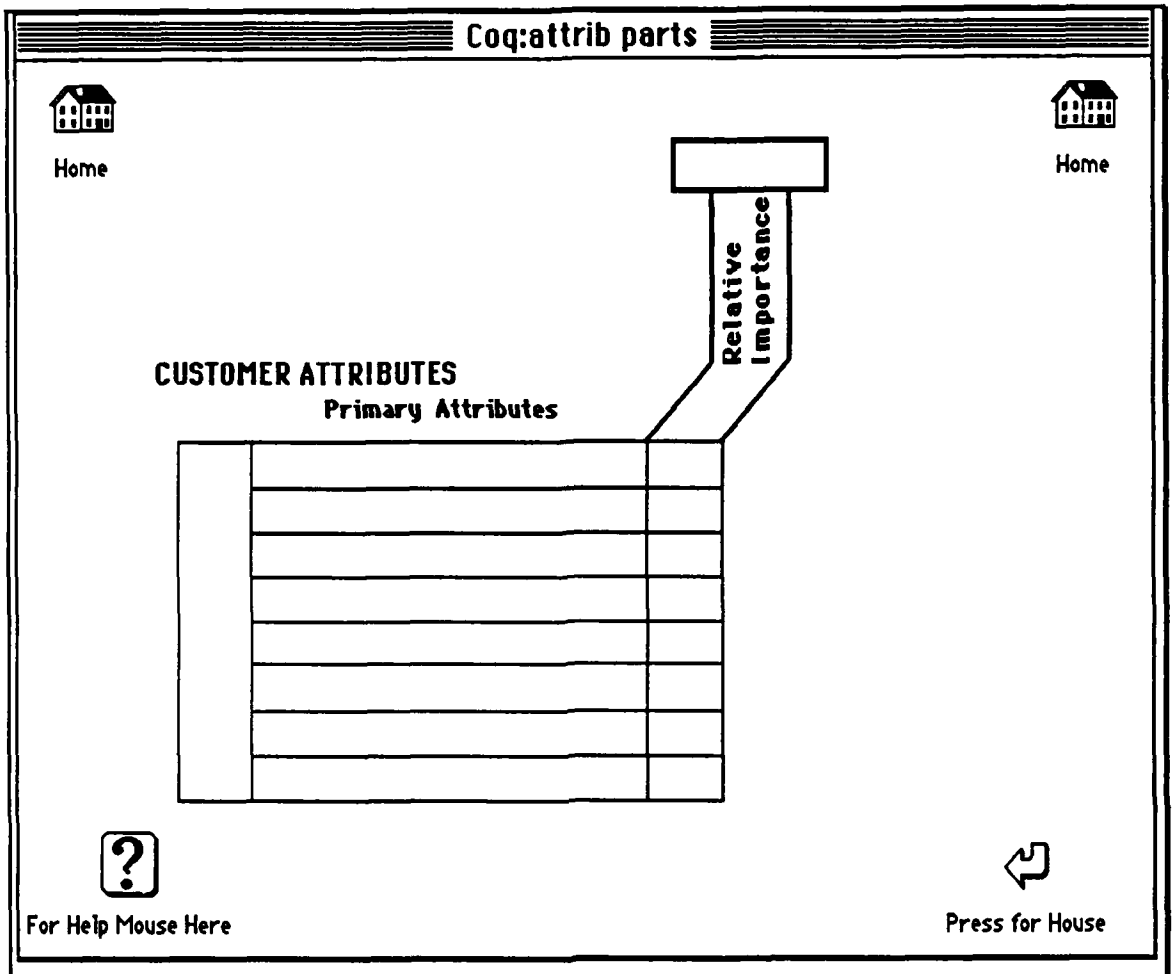


Figure 7.4 First Screen for Acquiring Customer Attributes

This card (node) has certain features common to all cards: clicking on the arrow in the lower right hand corner returns control to the parent of the current card; clicking on the little houses return to the root node; help, of course, is available by clicking on the lower left hand button. As explained at the help node, the user is to select one of the eight slots. This is intended to encourage a hierarchical decomposition of the customer attributes: it is easy to define up to 8 primary attributes. Each of these may be readily decomposed into 8 secondary attributes, etc. With effort, users may circumvent the 8 attribute limit, but the hierarchical approach with no more than 8 primary attribute and no more than 8 sub-attributes for each attribute is easier for the user to define than a flat approach with many attributes or sub-attributes at the same level.

When an empty slot is selected, the system creates a new node (card). A hard link is established between the slot and the new node, and selecting that slot again allows editing

the attribute. The node for defining/editing the attribute currently has the appearance shown in Figure 7.5.

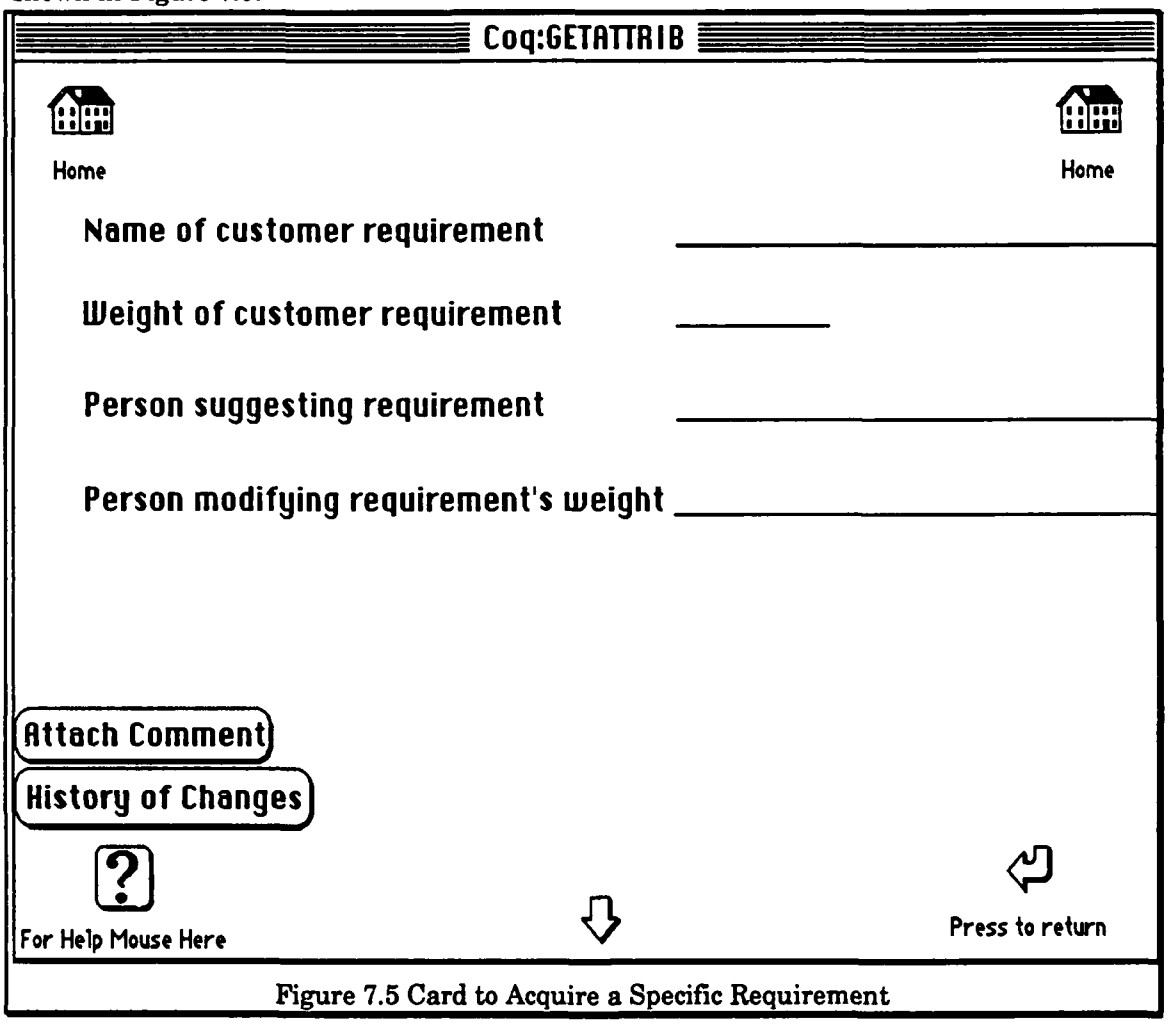


Figure 7.5 Card to Acquire a Specific Requirement

This card, created as needed, illustrates a few of the GDSS features of Coq. The non-disruptive annotation feature is provided by the "Attach Comment" command (called a **button** in Hypercard). Each user is allowed to attach one or more comments. Features for editing comments, for viewing other users' comments, etc. are possible.

The second GDSS feature is the History. A record is kept of every modification and the person responsible for the modification.

The downward pointing arrow at the bottom of the screen, the first time it is activated, adds a node like Figure 7.4. to the network, and creates a hard link to the node (thereby re-defining the function of the arrow from a create function to a transition function.) The result is an innovative use of self-modifying software, but one in which the self-

modifications are carefully controlled. The new card (Figure 7.4) will have the customer requirement of Figure 7.5 printed vertically in the box on the left-hand side, and the user will then be able to add secondary requirements. As each secondary requirement is added, a new card like Figure 7.5 will be created and hard linked to the overview of secondary requirements associated with its primary requirement.

Note, finally, that this process is recursive, and is limited only by the storage capacity of the secondary storage medium: the hierarchy may be easily extended to an arbitrary depth. After the system has been used, the structure chart might look like Figure 7.6.

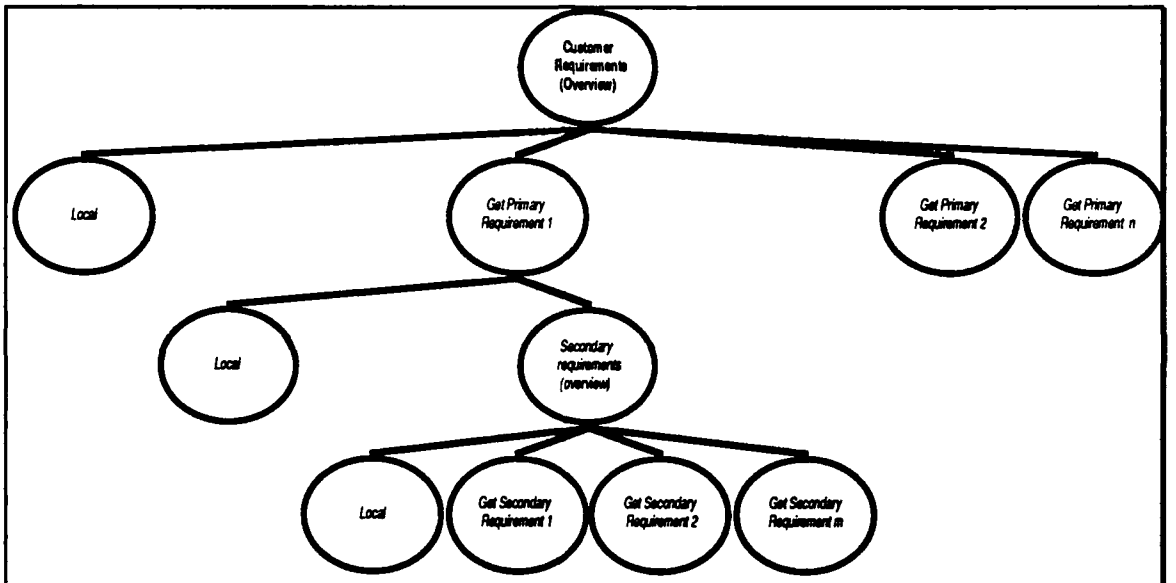

Figure 7.6 Possible Structure Chart for Customer Attribute Acquisition Chart Sub-tree

The process for acquiring technical specifications is similar, although somewhat different information is appropriate for technical specifications, as shown in Figure 7.7. In particular, quantitative targets, cost, and technical difficulty are acquired, in addition to relative importance. If used in a competitive situation, estimated values for competitors may be acquired and stored as well. Again, self-modifying software and recursion are used to make the "knowledge acquisition" process as painless as possible for the user.

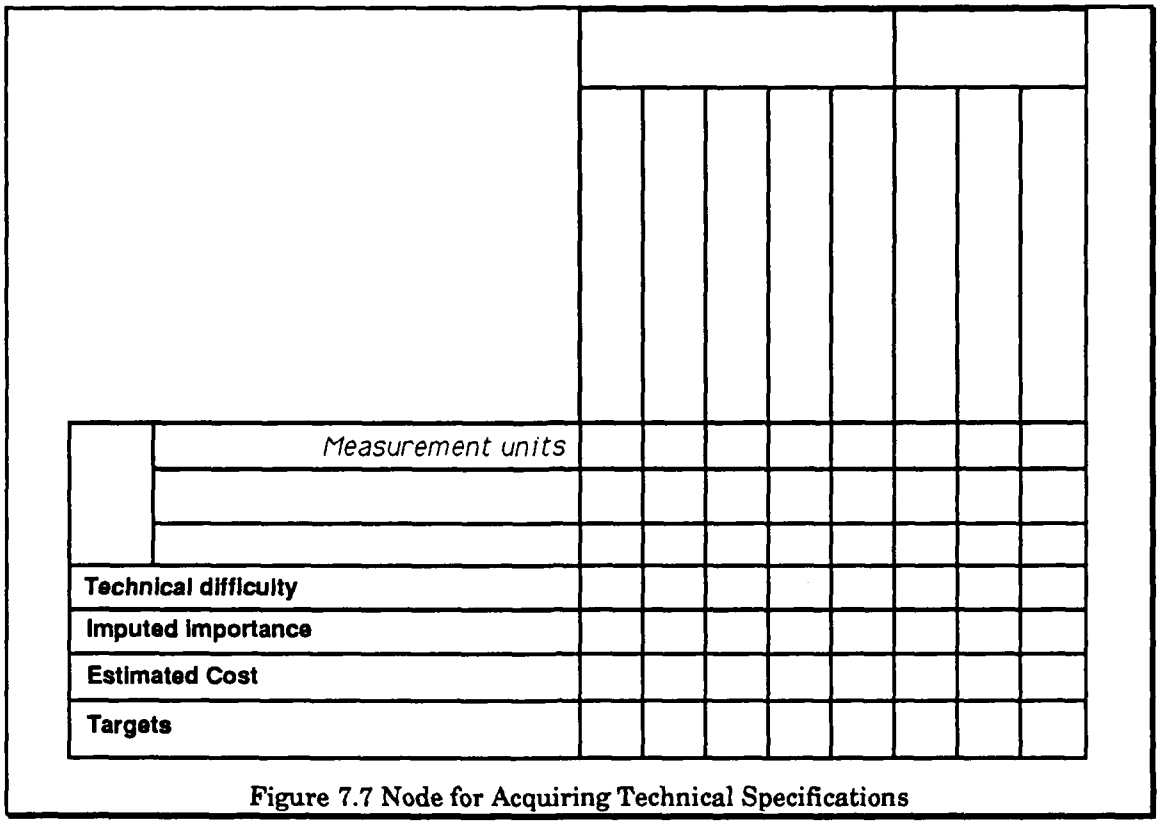


Figure 7.7 Node for Acquiring Technical Specifications

## VIII. SUMMARY

The basic mathematical theory of multi-person multi-attribute decision making developed in this paper shows the applicability of the Taguchi formalism. This formalism lends itself to the House of Quality paradigm for "knowledge acquisition" of the parameters that are required by the formalism. Finally, the process of turning qualitative user requirements into a functional system that addresses those requirements is not only amenable to automation, but requires automation in order to minimize the costs of the communications, including intertemporal communications, among the responsible decision makers.

This automation and communication are achieved by the City of Quality, or Coq system, developed using Hypertext methods. Innovative informatics involving self-modifying software were developed. The result is a powerful platform for group decision support of total quality and unified life-cycle engineering.

As a side benefit, for purposes of management of this (and future) hypertext projects, rough guidelines were established for hypertext projects, guidelines that can be extended as hypertext becomes an increasingly important tool for software development and prototyping.

## IX. RECOMMENDATIONS

The design of the Coq system has been described; implementation has been started, but is not yet complete. In developing Coq, it was discovered that hypertext, originally intended to allow an author to prepare a static but multi-dimensional document, is awkward for linking data between different parts of the document. Currently, this is addressed by building complex data structures; however, this solution has an adverse impact on maintainability of the system.

A second limitation of the implementation of hypertext used is based on the expectation of either limited user input, or input by a sophisticated user; hence, the platform has no provision for error trapping. As an example of the impact of this omission, should the user not understand that a numerical weight is needed, and instead provide a verbal description (or make a typo) the error will not be detected until a calculation is made with the data. The error will break out of Coq and into native Hypercard, where the user must figure out what to do. Since the error message may be delivered long after the error was made, this could prove very frustrating for the novice or casual user.

Partial solutions include developing an algorithm for distinguishing between valid numbers and non-numeric data, or performing an immediate calculation, so at least the error message (perhaps uninformative) will be more closely related to the action which produced it.

A major extension planned that will address both these limitations is to link all cards to a relational database, which will make data transfer relatively painless, as well as providing better security and integrity.

In its current form, the system is limited to sequential multi-user access. This limitation is partly the result of the current hardware platform, which is a stand-alone machine, and partly the result of limitations on the time available for system development. The system can be upgraded to concurrent access once a LAN platform and additional software tools become available.

In developing the system, a good start was made at developing project management tools for Hypertext projects. These need further testing and development to form a solid foundation for structured development of Hypertext.

The most important direction for further research, however, is to actually field test the system: this will conclusively identify deficiencies, strengths and potentials.

## REFERENCES


Akscyn, Robert M.; McCracken; Yoder, Elise A. KMS: A distributed Hypermedia System for Managing Knowledge in Organizations. Communications of the ACM; July, 1988; 31(7): 820—835.

Arrow, Kenneth J. Social Choice and Individual Values, 2nd Ed. Cowles Foundation for Research in Economics at Yale University: Yale University Press; 1963.

Charnes, A.; Cooper, W. W. Management Models and Industrial Applications of Linear Programming. New York: Wiley; 1961.

Currim, Imran S.; Sarin, Rakesh K. A Comparative Evaluation of Multiattribute Consumer Preference Models. Ma..agement Science; 1984; 30(5).

De Vera, Dennis; Glennon, Tom; Kenny, Andrew A.; Khan, Mohammad A. H.; Mayer, Mike. An Automotive Case Study. Quality Progress; 1988: 35—38.

Deming, W. Edwards. Out of the Crisis. Cambridge, MA. Massachusetts Institute of Technology; 1986; ISBN: 0-911379-01-0.

Duke, D. J.; Nijssen, G. M.; Twine, S. M. (Department of Computer Science, University of Queensland, St. Lucia, Australia). The Entity-Relationship Data Model Considered Harmful. Sixth Symposium on the Empirical Foundations of Information and Software Science; 1988; Atlanta, Georgia; 1988.

Elmaghraby, Salah E.; Fathi, Yahya; Ferrell, Jr., William G. (North Carolina State University). The Taguchi Approach to Quality Control and Enhancement: A Primer. Raleigh, NC; 1986; ARO 22835.1-MA.

Garvin, David A. Competings on the eight dimensions of quality. Harvard Business Review; November-December, 1987: 101—109.

Hauser, John R.; Clausing, Don. The House of Quality. Harvard Business Review; 1988; 66(3): 63—73.

Keeney, Ralph L.; von Winterfeldt, Detlof. On the Uses of Expert Judgment on Complex Technical Problems. IEEE Transactions on Engineering Management; 1989; 36(2).

Luce, R. Duncan; Raiffa, Howard. Games and Decisions. New York: John Wiley & Sons; 1957; ISBN: 0 471 55341 7.

Marschak, Jacob; Radner, Roy. Economic Theory of Teams. New Haven//London: Yale University Press; 1972; c1972. (Cowles Foundation for Research in Economics at Yale University Monograph; v. 22); ISBN: 0-300-01279-9.

Michie, D. Inductive Rule Generation in the Context of the Fifth Generation. Proceedings of the International Machine Learning Workshop; 1983; 1983: 65—70.

Pyeatt, Robert; Wolfe, Michael. PEG: The Portable Entrepeneurs' Group Decision Support System. CORS/ORSA/TIMS Joint National Meeting; 1989; Vancouver, BC; 1989.

Ross, Philip J. The Role of Taguchi Methods and Design of Experiments in QFD. Quality Progress; June, 1988.

Smith, John B.; Weiss, Stephen F. An Overview of Hypertext. Communications of the ACM; July, 1988; 31(7): 816—819.

Tse, Edison; Cralley, William E. (Stanford University// Institute for Defense Analysis). Management of Risk and Uncertainty in Product Development Processes; 1988.

Vaughan, Tay. Using Hypercard®. Carmel, Indiana: Que® Corporation; 1988; ISBN: 0-88022-340-5.